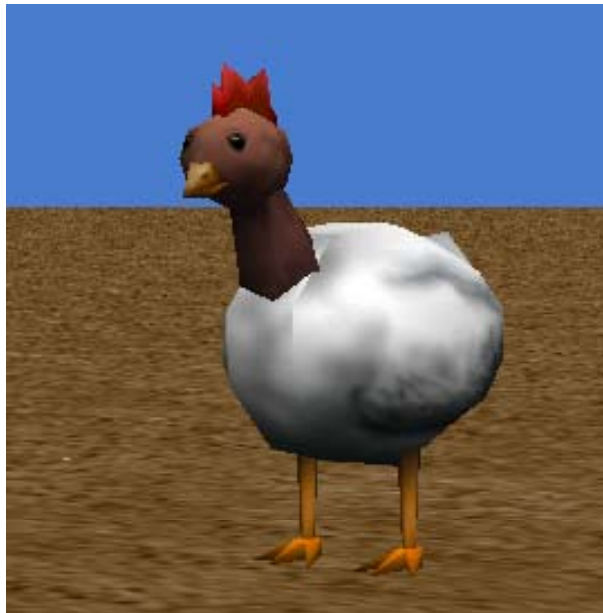


# Making Events Better: Restricting an Event with a Conditional



By Henry Qin, edited by Jenna Hayes  
Under the direction of Professor Susan Rodger  
Duke University, August 2008

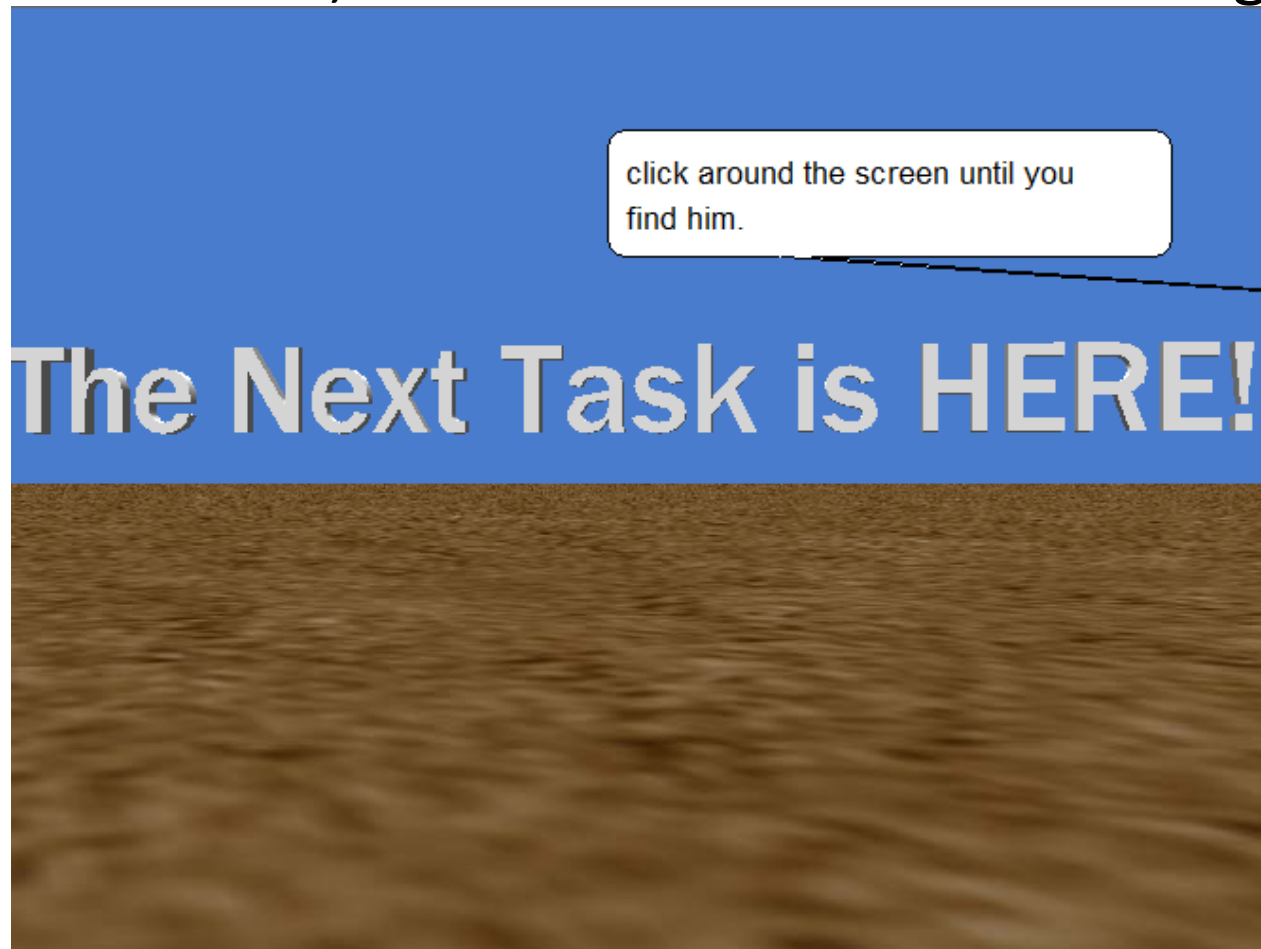
# Getting Started

Download the starting world that goes with this tutorial. The world contains the beginnings of a game in which you have to find a chicken's invisible chick, click on it, and then press **enter** to go to the next task. The problem is, if you press **enter** *before* finding the chick, it will go to the second task anyway. This tutorial will teach you how to fix this problem, and make your chicken game work like a real game, so that you can't move on until you've finished the first task.



# Testing it Out

Try playing the world and pressing **enter** without even trying to find the chick. Your camera will go to the next task, but the chicken will not even be finished telling you about the first task, which will result in something like this:



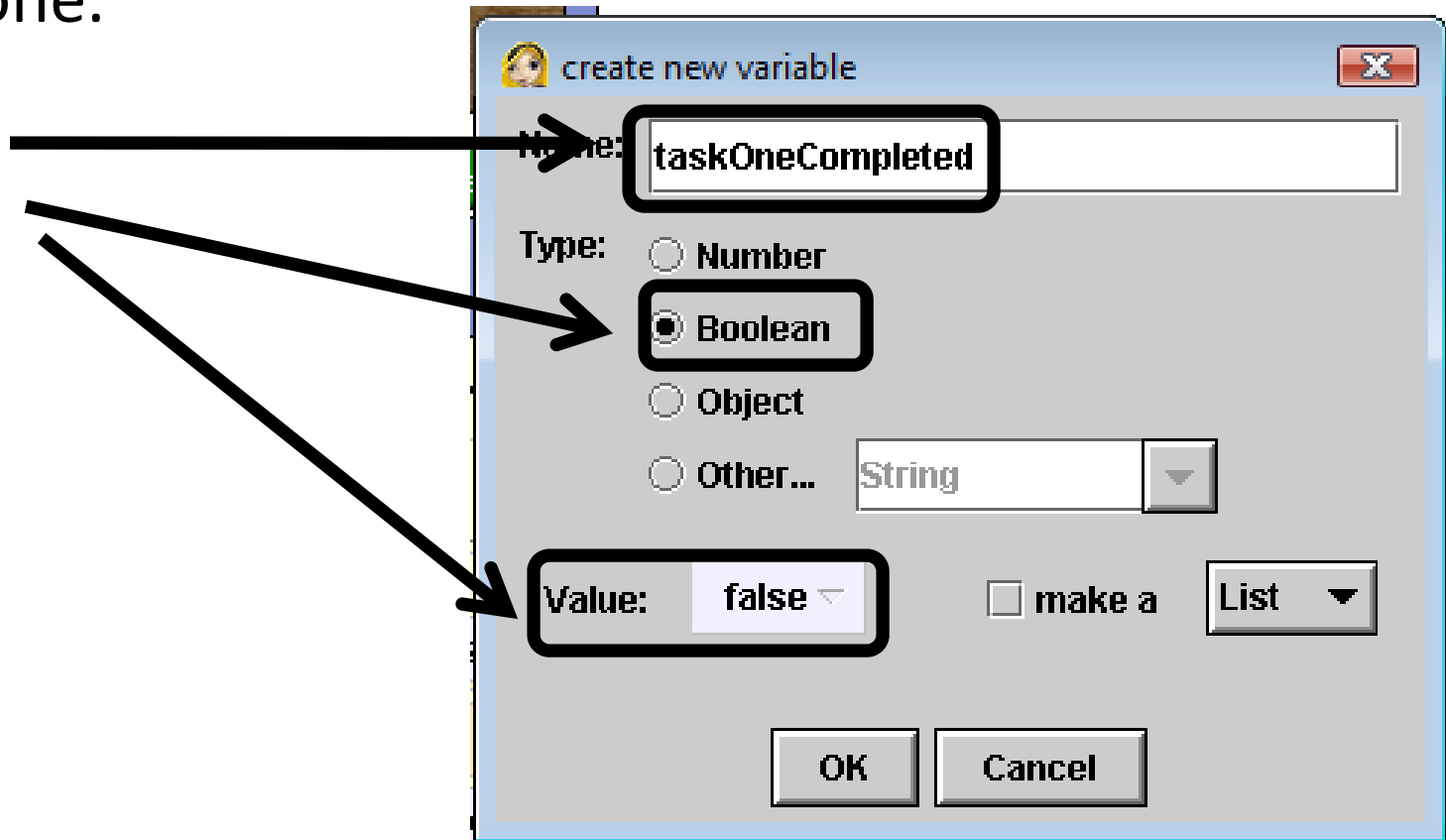
# Creating a Boolean Variable

First we need to create a **boolean** (true or false statement) **variable** that will be true if the first task is finished, and false if it is not. Click on **world** in the object tree and then go to the **properties** pane and click on **create new variable**.



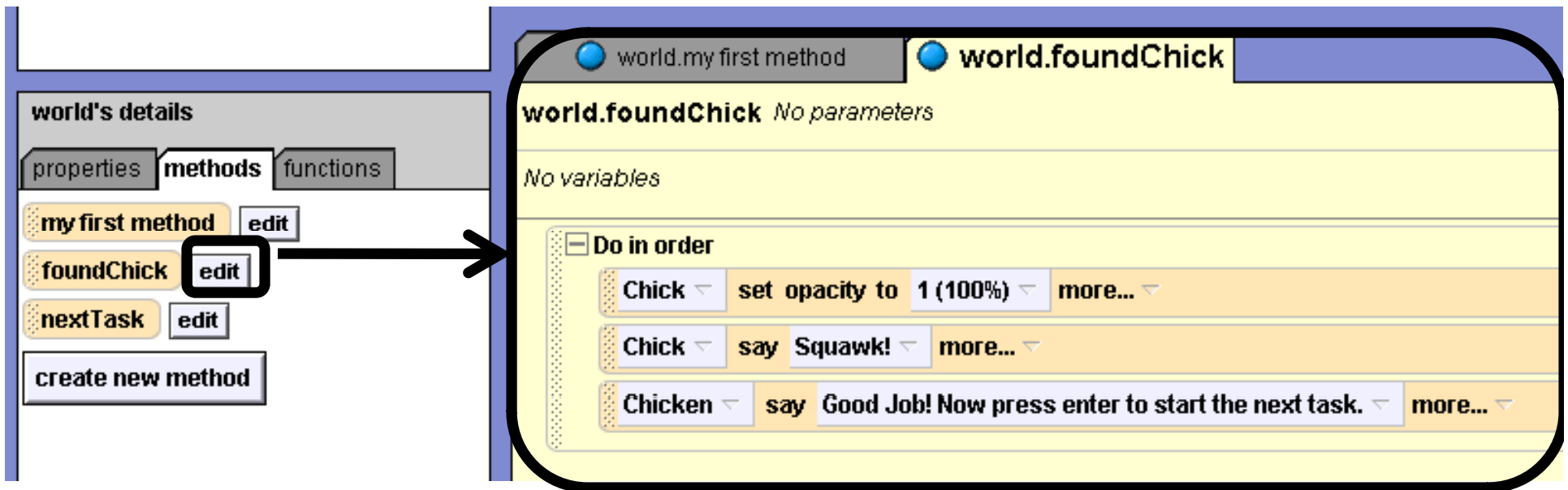
# Creating a Boolean Variable

When the variable box pops up, name your variable **taskOneCompleted**. Make sure you have selected **Boolean**, and set your variable to **false**, because the variable should not be **true** until the first task is completed. Click **Okay** when you're done.



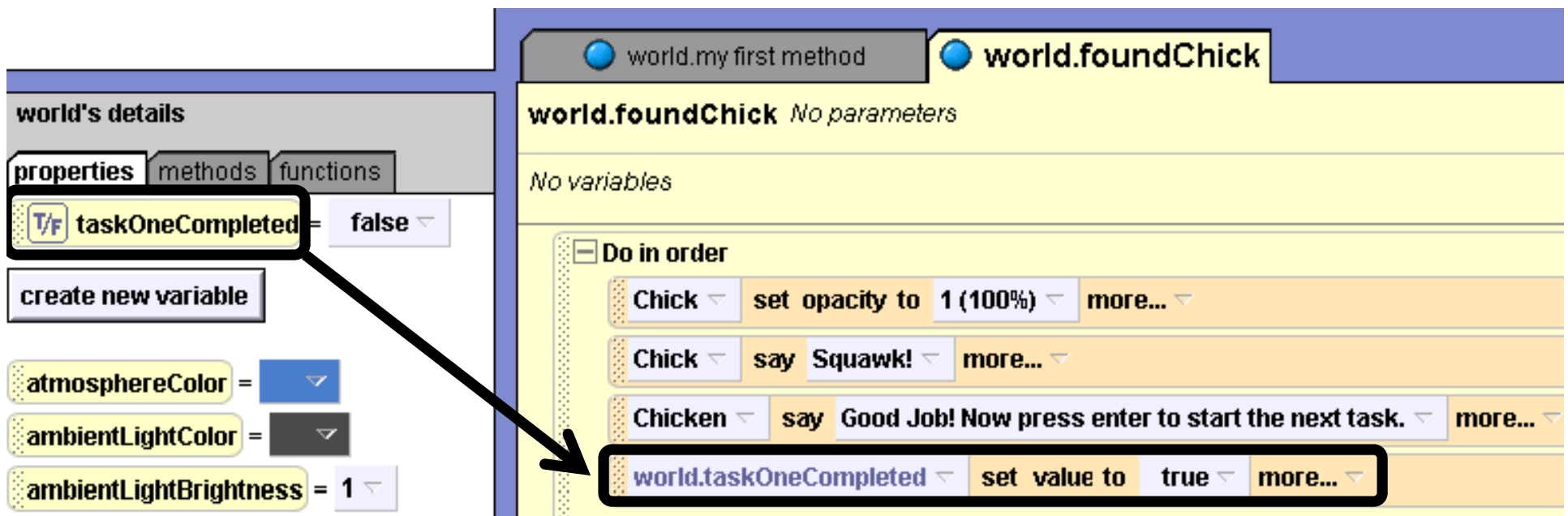
# Setting the Variable

Now, we need to include code somewhere that changes **taskOneCompleted** to true when the chick is clicked on. For this, we want to look at the **foundChick** method, because as you can see in your events editor, this method runs when the chick is clicked on. Go to the world's **methods** pane and click on **edit** next to **foundChick**.



# Setting the Variable

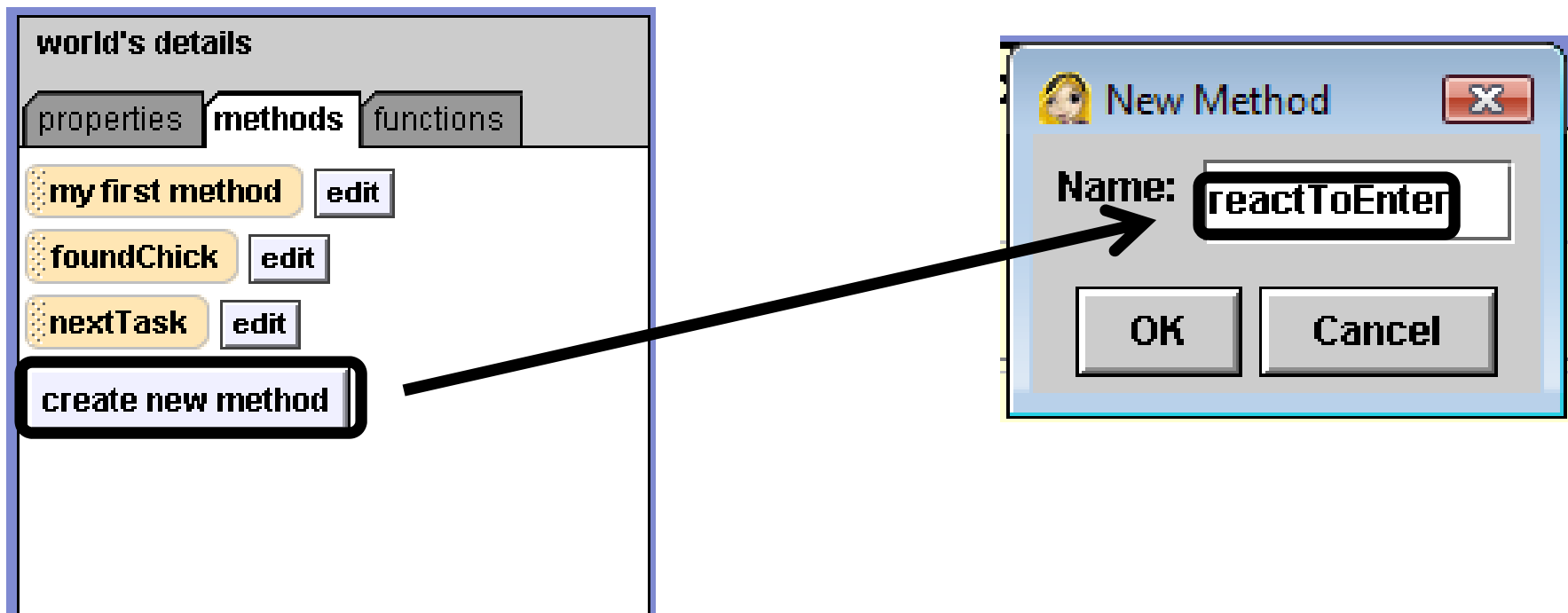
Go back to the **properties** pane and find **taskOneCompleted**. Drag and drop it to the bottom of your **foundChick** method. Select **true**.



Now **taskOneCompleted** will be set to **true** when the chick is clicked on.

# Using the Variable

Now we need to use the information that the boolean variable tells us; we need a method that tells Alice to only go to the second task if **taskOneCompleted** is **true**. We will create a new method for this. Go back to the **methods** pane, and click on **create new method**. Name it **reactToEnter**.





# Building reactToEnter

First, drag an **If Else** statement into **reactToEnter** and set it to **true**.

The image shows the Scratch IDE interface for editing the **world.reactToEnter** method. The top bar shows three tabs: **world.my first method**, **world.foundChick**, and **world.reactToEnter**. The **world.reactToEnter** tab is active, showing the method name and "No parameters". Below this, it says "No variables". On the right side of the workspace, there are two buttons: "create new parameter" and "create new variable".

The workspace contains an **If Else** block. The **If** section is set to **true** and contains a **Do Nothing** block. The **Else** section also contains a **Do Nothing** block. A black arrow points from the **If/Else** block in the bottom palette to the **If Else** block in the workspace.

The bottom palette shows various control blocks: **Do in order**, **Do together**, **If/Else** (highlighted with a black box), **Loop**, **While**, **For all in order**, **For all together**, **Wait**, **print**, and a comment block.

# Building reactToEnter

Now go back to the **properties** pane and find **taskOneCompleted**. Drag and drop it over where your **If Else** statement says **true**.

The screenshot shows a programming environment with three tabs at the top: 'world.my first method', 'world.foundChick', and 'world.reactToEnter'. The 'world.reactToEnter' tab is active, showing a script area with the text 'world.reactToEnter No parameters' and 'No variables'. There are two buttons on the right: 'create new parameter' and 'create new variable'. The script area contains an 'If/Else' statement. The 'If' condition is 'taskOneCompleted', which is highlighted with a green box and a black circle. The 'Else' condition is 'Do Nothing'. The bottom toolbar includes buttons for 'Do in order', 'Do together', 'If/Else', 'Loop', 'While', 'For all in order', 'For all together', 'Wait', 'print', and a comment icon.

# Building reactToEnter

Go back to the **methods** pane and find **nextTask**. Drag and drop it into your **If Else** statement.

The image shows a Scratch workspace with three tabs: **world.my first method**, **world.foundChick**, and **world.reactToEnter**. The **world.reactToEnter** tab is active, showing a method with **No parameters** and **No variables**. On the right, there are buttons for **create new parameter** and **create new variable**. The main workspace contains an **If** statement with the condition **world.taskOneCompleted**. The **Do Nothing** block in the **If** branch is highlighted with a green box and labeled **nextTask**. A black arrow points from the **nextTask** label to the **Do Nothing** block. The **Else** branch is also visible with a **Do Nothing** block. The bottom toolbar shows various control blocks: **Do in order**, **Do together**, **If/Else**, **Loop**, **While**, **For all in order**, **For all together**, **Wait**, **print**, and a comment block.

# Changing the **Enter** Event

Now when we hit **Enter** instead of doing **nextTask**, we want Alice to do **reactToEnter**. Drag and drop **reactToEnter** from the **methods** pane to where it says **nextTask** on the **Enter** event.

The image shows a software development interface with two main panels. On the left, the 'world's details' panel has tabs for 'properties', 'methods', and 'functions'. The 'methods' tab is active, displaying a list of methods: 'my first method', 'foundChick', 'nextTask', 'reactToEnter', and 'create new method'. Each method has an 'edit' button. The 'reactToEnter' method is highlighted with a black box. On the right, the 'Events' panel has a 'create new event' button and a list of event triggers. The third event is 'When Enter is typed, do', which is also highlighted with a black box. A large black arrow points from the 'reactToEnter' method in the left panel to the 'do' field of the 'When Enter is typed' event in the right panel.

**world's details**

properties methods functions

my first method edit

foundChick edit


nextTask edit

**reactToEnter** edit

create new method

**Events** create new event

When the world starts, do world.my first method ▾

When  is clicked on Chick ▾, do world.foundChick ▾

When Enter ▾ is typed, do **world.reactToEnter** ▾

# Testing it Out

Now play the world, and try to press **Enter** without first clicking on the invisible chick. It won't work! Only after you have found the chick will you be able to continue.

